УДК 681.3.06

Л.П. БАБЕНКО

ИНЖЕНЕРИЯ ТРЕБОВАНИЙ И СЕМАНТИЧЕСКИЙ ВЕБ[1]

Аннотация. Рассмотрены современные методы ориентации процессов инженерии требований на понимание их смысла человеком и предложен подход для трансформации этих требований в их семантические аннотации, пригодные для манипуляций ими в среде семантического веба.

Ключевые слова: инженерия требований, семантические аннотации, семантический веб, спецификация примерами, коммуникации в команде.

ВВЕДЕНИЕ

Главным (если не единственным) источником знаний о любых современных программных продуктах являются требования к их разработке, на базе которых создаются все вторичные документы, описывающие различные аспекты их функционирования. Именно требования к разработке программного продукта (в частности, веб-сервиса), если они соблюдены, определяют его функциональные, организационные, исполнительские и коммуникационные свойства. Для повторного использования программных продуктов требования необходимо трансформировать в аннотированные спецификации семантических свойств, которые могут являться базой хранения и поиска последних в различных хранилищах (регистрах).

Разработчики программного обеспечения признают, что большинство недостатков программных продуктов обусловлено несовершенством процессов инженерии требований: ошибки, неточности, неполнота информации приводят к многочисленным переделкам, в результате чего срываются сроки разработки, увеличивается ее стоимость, а поставка заказчику системы с ошибками влечет для него риски тяжелых последствий и угроз.

Несовершенство процессов инженерии требований является следствием несовершенства языков их спецификации. Действительно, много лет разрабатываются стандарты таких языков, предназначенные для решения двух плохо совместимых задач:

— обеспечить заказчику программного продукта такое понимание требований, чтобы он мог засвидетельствовать их полноту и правильность;

— выполнить такую формализацию языка представления требований, чтобы его однозначно понимали разработчики программного продукта без привлечения заказчика.

Если в начале развития программной индустрии предметом поиска решения были формализованные методы абстрактного представления требований, основанные на аппаратах математической логики, алгебры и т.д., то в настоящее время осознана ключевая роль заказчика системы — непрофессионала в области компьютерных технологий, с которым следует общаться только на понятном ему языке.

Первым значительным шагом такой адаптиции требований к обычным понятиям заказчика, в частности из сферы бизнеса, стало появление и широкое применение языка UML (Unified Modelling Language — унифицированный язык моделирования), который впоследствии существенно потеснил язык BPMN (Business Processes Modelling Notation — нотация моделирования процессов бизнеса).

[1] Работа проведена в рамках Государственной программы фундаментальных исследований Национальной академии наук Украины.

Общим недостатком этих средств выражения требований является то, что их использование становится возможным, когда уже приняты определенные решения по проектированию будущей системы, а именно проведена крупноблочная алгоритмизация ее функций, определены объекты, которыми данные функции должны манипулировать, и состояния системы, влияющие на их поведение, а также события, изменяющие эти состояния, и т.п. Для решения таких задач необходима компетентность в разработке компьютерных систем, которую не имеют заказчики. На практике приведенные языки стали инструментом коммуникации для разработчиков различных этапов создания программ, но результат этой коммуникации — представление требований, не гарантировал их соответствия целям заказчика и его исчерпывающего понимания того, что именно для него будет строить разработчик.

Приведенные аргументы подвигли разработчиков к быстрейшему привлечению заказчиков к определению целей и ожидаемых результатов работы новой системы, а также к возможности непосредственно влиять на автоматизацию средств валидации продуктов разработки, т.е. соответствия программного продукта потребностям и пожеланиям заказчика. Ключевым условием для этого является создание современных моделей спецификации требований, обеспечивающих хорошую коммуникацию между всеми участниками (так называемыми актерами) процессов инженерии требований. Такие модели должны решать следующие задачи:

— обеспечивать надежность спецификации требований и ее понимание всеми актерами (заказчиками и разработчиками);

— поддерживать актуальность спецификации требований при всех будущих изменениях системы;

— при необходимости корректировки системы не требовать приложения бóльших усилий для внесения соответствующих изменений в требования, чем для непосредственной реализации изменений;

— являться критерием валидации создаваемых систем.

В настоящее время предложены принципиально новые подходы к инженерии требований и основанные на них технологии, ключевым отличием которых является улучшение коммуникации между актерами процессов инженерии требований как основы обеспечения качества разработок. Базой для этих подходов стала довольно неожиданная и простая идея — определять требования к новой системе с помощью примеров ее будущего применения для решения конкретных проблем пользователей. Эту методику назвали спецификацией требований примерами (Specification by example, SBE) [1], она обобщает такие подходы, как Test-driven development, Behavior-driven development, Agile Acceptance Testing, Executable Requirements [2–4], которые, по мнению автора настоящей статьи, являются доминирующими, в частности, при применении методов разработки Agile. Очевидное преимущество SBE — понимание спецификаций требований всеми актерами (людьми) создания программного продукта на всех стадиях.

Появление идей семантического веба [5] открыло новые возможности для автоматизации процессов создания программных систем, а именно включение в состав участников проекта не только специалистов, но и программ, в частности, так называемых веб-сервисов, все шире используемых в современных разработках. Однако для этого необходима формализация спецификаций требований до уровня их «семантизации», т.е. представления, позволяющего однозначно понимать их смысл не только разработчику (человеку), но и программе (сервису).

Далее рассмотрены современные методы ориентации процессов инженерии требований на понимание их смысла разработчиком и предложен поход для их трансформации в семантические аннотации, пригодные для манипуляций ими в среде семантического веба

КОНЦЕПТУАЛЬНЫЕ ОСНОВЫ МЕТОДИКИ СПЕЦИФИКАЦИИ ТРЕБОВАНИЙ ПРИМЕРАМИ

Целью применения методики [1] является улучшение коммуникаций, т.е. преодоление коммуникационных барьеров между всеми актерами разработки программного продукта. Они должны пользоваться одним языком и одинаково правильно понимать домен. При этом необходимо, чтобы общий язык проекта базировался на бизнес-ориентированном словаре для обеспечения однозначной связи для команды.

Базовое решение. Требования заказчика к разработке представляются как реальные примеры применения программного продукта, которые в то же время являются критериями приемки при поставке его пользователям и тестами при внесении будущих изменений. Такое представление дает преимущества всем участникам разработки, а именно:

— владельцы программных продуктов, бизнес-аналитики и менеджеры проектов получают уверенность, что разработчики и тестировщики понимают их правильно, а также имеют возможность следить за ходом разработки, определять конфликты между бизнес-правилами и требованиями при необходимости последующих изменений и экономить время на создание тестов приемки готового программного продукта;

— разработчики уже на начальных стадиях создания программного продукта избегают непонимания функций, требующихся заказчику, пробелов и несоответствий в требованиях и спецификациях, получают уверенность, что бизнес-аналитики действительно поняли специальные условия приложений, создают автоматизированные тесты, облегчающие взаимодействие всех актеров каждого этапа разработки при создании и сдаче кода;

— тестировщики имеют возможность влиять на процесс разработки и предотвращать повторные ошибки разработчиков, лучше понимать проблемную область, делегировать определенную часть рутинной работы по автоматизации тестирования разработчикам.

Общий язык актеров разработки как основа понимания. Спецификация требований примерами рассматривается как общий язык общения, понятный для всех актеров команды проекта на каждой стадии его жизненного цикла и во всех артефактах, которые получают или создают актеры (как заказчики, так и разработчики). Это аналог известного понятия ubiquitous language — термина, обозначающего общий язык коммуникации всех участников команды проекта создания программного продукта.

Концепция выполняемой спецификации. Спецификация требований примерами выполнима, отвечает на вопрос, «как проверить требование» тестами приемки готовой системы, если построить тесты выполнения этих примеров. При этом необходимо, чтобы тесты были полными, явно определенными, а их количество было конечным, выполнялось над реальными данными и за практически приемлемое время.

Тест требования должен проверять следующие три случая использования системы: нормальное; ненормальное, но понятное; ненормальное и непонятное. Последнее относится к действиям, которые вроде бы не должены осуществляться, но в реальности они все-таки могут произойти. Необходимо, чтобы такие моменты тестировщики определяли как случаи, которые могут привести ко взлому системы.

Автоматизация средств валидации без изменения спецификаций. Автоматизация запуска тестов приемки позволяет использовать их на всех стадиях разработки и контролировать степень готовности программного продукта.

Определение предназначения разрабатываемого программного продукта. Самым трудным в построении программного продукта является точное опре-

деление того, что именно строить. Прежде чем спрашивать пользователя о том, что должна делать система, которую он заказывает, надо понять, почему он ее заказывает, кто и как часто будет ее использовать. Далее необходимо знать приоритеты того или иного свойства проекта с точки зрения бизнес-пользователей, а также понять, какую информацию они хотят получать в результате работы системы, после чего можно сосредоточиться на работе. Спецификация требований примерами должна явно или неявно отражать цели заказчика.

Сотрудничество при создании спецификаций требований примерами. В создании таких спецификаций необходимо участие всех слоев актеров разработки. Например, заказчики имеют знания о проблемной области, к которой относится заказанное программное приложение, и могут передать их разработчикам; технические эксперты знают, как лучше использовать базовую инфраструктуру или как можно применить новые технологии; тестировщики знают, где могут возникнуть потенциальные проблемы, для предотвращения которых нужны специальные тесты приемки.

Отображение нефункциональных требований. К ним относят такие характеристики, как производительность, удобство использования или время отклика и т.д. Общей характеристикой большинства нефункциональных требований является то, что каждое из них определяет точку перехода продукта от неиспользуемого (бесполезного для бизнеса) к используемому (полезному), например, создающему конкурентные преимущества, которые будут влиять на маркетинг.

Принципы преобразования примеров в тесты. При формализации примеров в тесты необходимо идентифицировать отдельные бизнес-правила и указать определенный тест для каждого из них. Бизнес-правило может использовать несколько тестов, тогда выявляют его параметры, которые отличают эти примеры (и соответствующие тесты). При этом пограничные и пороговые значения для правила полезно отнести к параметрам, чтобы их можно было в дальнейшем легко изменять и хранить.

СЕМАНТИЧЕСКОЕ АННОТИРОВАНИЕ СПЕЦИФИКАЦИИ ТРЕБОВАНИЙ ПРИМЕРАМИ (SBE) В СЕМАНТИЧЕСКОМ ВЕБЕ

Семантический веб (всемирная сеть) представляет собой информацию, снабженную комментариями, помогающими разработчикам и созданным ими программам понять смысл данной информации и, следоввтельно, манипулировать ею без обращения за разъяснениями к ее авторам. Такие комментарии называют семантическими аннотациями.

В настоящее время для построения универсальных средств представления семантических аннотаций разработаны базовые (не зависящие от домена информации) средства, например, формальные модели знаний (онтологии и др.), языки представления этих моделей (OWL, RDF, RDFS и др.) и инструменты их реализации. Развивая эти наработки, многие профессионалы заняты поиском моделей семантических аннотаций, конкретизированных для соответствующих им предметных областей (доменов). Успешность их использования зависит от стандартизации специализированных семантических аннотаций и их признания заинтересованными специалистами.

Семантические аннотации предназначены для записи толкования смысла аннотированного объекта его авторами в виде, доступном для понимания другими лицами, в качестве основ для применения автоматических инструментальных средств для последующей трансформации объектов аннотирования в исполняемые коды, а также для организации регистров (библиотек) соответствующих объектов аннотирования и дальнейшего семантически осмысленного поиска таких объектов для их интеграции (в том числе, автоматической) в создаваемые но-

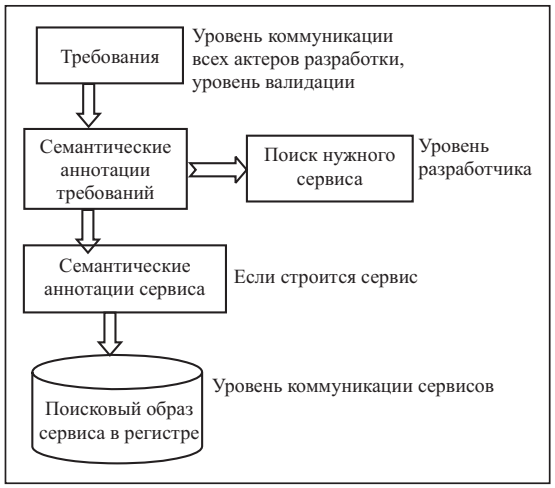

Рис. 1. Уровни спецификации требований к программному продукту

вые программы. Эти действия можно совершать, используя один и тот же текст семантической аннотации, однако с определенной для каждого процесса степенью лаконичности и выразительности, обеспечивающих их эффективность.

Объектом настоящего исследования являются требования к разработке программных систем, понимание которых нужно обеспечить на всех уровнях работы (рис. 1).

В результате многолетних безуспешных попыток создать единую нотацию для всех уровней разработчики пришли к выводу, что для для каждого уровня нужно иметь свою нотацию, максимально удобную для соответствующих ему актеров. Так, на верхнем уровне (см. рис. 1) целесообразно использовать нотацию SBE. Уровень разработчика должен отражать требования к основным решениям системы, т. е. состоять из традиционных утверждений «система должна», но представленных в формализованном виде, однозначно понятном всем разработчикам: бизнес-аналитикам, инженерам предметной области, проектировщикам, кодировщикам, тестировщикам, документаторам. Языком спецификаций этого уровня могут являться известные нотации UML и BPML. Однако при этом необходимо определить правила трансформации с уровня на уровень, которые не позволяли бы искажать смысл требований. Для этого вначале нужно создать стандартизированные и взаимно согласованные модели, соответствующие каждому уровню понимания, а именно спецификации предметной области и требований к применению, а также веб-сервисов и процессов инженерии требований.

Поскольку в настоящее время механизм онтологий [5, 6] — наиболее распространенный инструмент моделирования смысла знаний, такое согласование необходимо отражать в соответствующих онтологиях (рис. 2).

Термин «семантизация» означает выявление смысла, значения языковой единицы, а «семантизация требований» — их представление в нотации, которая однозначно понимается актерами соответствующего уровня. При этом будем считать, что трансформация представления требований с уровня на уровень рассматривается как отдельный процесс жизненного цикла инженерии требований.

Проведенный анализ современных подходов к построению семантических аннотаций артефактов сервис-ориентированных приложений в среде семантического веба [7–13] позволяет констатировать, что фактически преимущественным направлением семантического обогащения всех онтологий (см. рис. 2) является включение соответствующих концепций известной модели систем переходов в состояния. Отметим, что базовыми концепциями последней наряду с объектами и их атрибутами есть состояния

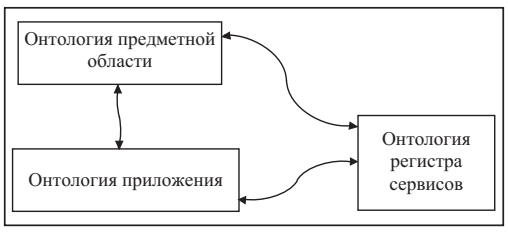

Рис. 2. Онтологии, используемые в инженерии требований

объектов, их действия, изменяющие эти состояния, цели и возможности процессов, подпроцессов, действий и задач. Включение их в верхние уровни описанных онтологий приводит к общей системе понятий, характеризующих три типа объектов моделирования: проблемную область (домен), бизнес-процессы и семантические веб-сервисы. Таким образом, семантизация требований примерами заключается в построении семантических аннотаций к каждому из примеров в терминах концептуальных единиц соответствующих моделей.

ПОДХОД К ТРАНСФОРМАЦИИ ТЕСТОВ SBE В СЕМАНТИЧЕСКИЕ АННОТАЦИИ ТРЕБОВАНИЙ

Представим процесс семантизации следующими шагами.

1. Экстракция концептуальных единиц из примеров. Выделение текстовых единиц, не являющихся литералами, и определение категории сущностей, которые они идентифицируют (объекты и понятия домена, отношения между ними, характерные действия, ситуации (состояния), события, данные и т.п.).

2. Онтологизация концептуальных единиц. Этим термином будем обозначать установление соответствия между идентификаторами концептуальных единиц примеров и онтологии домена, замену идентификаторов примеров на идентификаторы онтологии домена и маркировку каждой концептуальной единицы категориями онтологии спецификации требований. Последняя включает такие понятия [8], как цель, процесс, объект его действия и атрибуты, а также состояние, событие, актер, эффект и функция процесса.

3. Составление семантической аннотации спецификации требований в терминах онтологизированных концептуальных единиц. Такое представление обеспечивает однозначность толкования аннотаций в рамках выделенных предметных областей и позволяет строить поисковые образы объектов хранения регистров программных продуктов в онтологизированных терминах, облегчающих их последующий поиск.

Процесс семантизации можно автоматизировать с помощью специальной службы — лексикона, функцией которой является удержание всех лиц, которые обращаются к онтологии, в поле нормализованной лексики, идея которой представлена в [14].

Главной функцией лексикона является учет всех участвующих в данной онтологии имен концептов, их отношений, атрибутов и т.п. Иначе говоря, он должен вести нормализованную терминологическую базу взаимодействия актеров с онтологией. При этом организацию лексикона можно совмещать с процессом создания онтологии. Каждый термин, помещаемый в лексикон, снабжается атрибутами, указывающими его роль в онтологии (имена концепта, отношения, атрибута отношения, а также имя непосредственного родителя концепта).

Рассмотрим проблему синонимии. Введения аппарата поддержки синонимии для терминов есть способ удовлетворить персональные вкусы или собственное понимание семантики терминов, в частности для новых предметных областей или при переводе с иностранных языков. Вместе с этим проблема синонимии является наиболее острой в информационных системах, поэтому на ее решении в данной разработке остановимся детальнее.

В [14] предложено для каждого концепта онтологии иметь единственное эталонное представление его имени, которое указано в онтологии, при этом в лексиконе для него вводится специальный атрибут — «синоним» со множественными значениями, представляющими синонимы эталонного имени.

Лексикон должен оказывать справочные услуги, определяя для запрашиваемого термина его принадлежность к указанной онтологии, его роль в ней (концепт, его синоним и соответствующий ему эталонный термин, отношение, атри-

бут и т.п.) или отсутствие термина в онтологии. Таким образом, лексикон является фильтром при употреблении ненормализованной лексики. Лексикон можно реализовать как реляционную базу данных.

ЗАКЛЮЧЕНИЕ

Предложенный поход к трансформации ориентированных на разработчика требований в их семантические аннотации позволяет совместить возможности эффективной коммуникации заказчиков и разработчиков программной продукции с возможностями автоматизированной манипуляции программными продуктами средствами семантического веба.

СПИСОК ЛИТЕРАТУРЫ

1. Adzic G. Specification by example. How successful teams deliver the right software. — New York: Manning Publications, 2011. — 295 p.
2. Chelimsky D., Astels D., Zach D., Hellesoy A., Helmkamp B., North D. The RSpec Book: Behaviour-Driven Development with Rspec, Cucumber, and Friends. (First ed.) . — Raleigh (North Carolina); Dallas (Texas): The Pragmatic Bookshelf, 2010. — 448 p.
3. Bolton M. Acceptance tests: let's change the title, too. — http://www.developsense.com/blog/2010/08/acceptance-tests-lets-changethe-title-too/.
4. Adzic G. Bridging the communication gap: Specification by example and agile acceptance testing. — London: Neuri Limited, 2009. — 282 p.
5. Castañeda V., Ballejos L., Caliusco M.L., Galli M.R. The use of ontologies in requirements engineering // Global Journal of Research and Development. — 2010. — **10**, N 6. — P. 2–8.
6. Ellis K. Three golden rules of requirements repositories. — http://blog.enfocussolutions.com/Powering_Requirements_Success. — Oct 28, 2013.
7. Siegemund K.I., Thomas E.J., Yuting Zhao, Pan J., Assmann U. Towards ontology-driven requirements engineering // 10th International Semantic Web Conf. (ISWC 2011). — Bonn (Germany), 2011. — P. 537–561.
8. Fritsch D., Lehmann J., Lauenroth K., Lohmann S., Riechert Th., Dietzold S. The SoftWiki ontology for requirements engineering (SWORE). — http://ns.softwiki.de/req).
9. Бабенко Л.П. Характеристический анализ современных подходов к спецификации семантики программ // Труды второго симпозиума «Онтологическое моделирование». — Казань, 11–12 октября 2010 г. — М.: ИПИ РАН, 2011. — С. 270–285.
10. Web service description language. W3C recommendation. — http://www.w3.org/TR/2007/REC-wsdl20-20070626.
11. Semantic annotations for WSDL and XML schema. W3C recommendation. — http://www.w3.org/TR/sawsdl/.
12. Web service modeling ontology. — WSMO http://www.w3.org/Submission/WSMO/.
13. Kohlborn T., Korthaus A., Chan T., Rosemann M. Identification and analysis of business and software services — a consolidated approach // Transactions on Services Computing. — 2009. — **2**, N 1. — P. 81–96.
14. Бабенко Л.П., Поляничко С.Л. Методы нормализации знаний в инфраструктуре разработки программ // Кибернетика и системный анализ. — 2006. — № 1. — С. 167–173.

Поступила 06.05.2015